OpenXT Project in 2016

Christopher Clark
BAE Systems

Xen Developer Summit, 25-26th August, 2016
Presenter: Christopher Clark

Xen affiliations:

- **BAE Systems** and the **OpenXT Project** - since January 2016.
- [ non-Xen engineering work, 4 years ]
- Citrix Systems
- XenSource Inc.
- Intel Corporation
- Cambridge University Computer Laboratory

Projects:

- **OpenXT** - XenClient XT - XenClient - XenServer - Xen hypervisor

Roles:

- **Interoperability Architect**
- Release Manager
- Principal Engineer
- Graduate Student

Focus of work in 2016: **Governance** for the OpenXT Project; **Security review** of OpenXT software.
Outline

● **Introduction** to OpenXT

● **Distinguishing aspects** of OpenXT

● **Current activity** within the OpenXT Project
  ○ Recent developments
  ○ Work in progress
  ○ Roadmap items
OpenXT: motivation*

- Advance progress towards **more secure computing** through a modern, practical, open system architecture.

- **Provide a common platform of software to build upon,** prioritizing extensibility, adaptable to diverse use cases, to promote collaboration and sharing of common components.

- Support building usable systems that are **robust to partial compromise**.

*Note: these are the opinion of the presenter*
OpenXT nutshell

- **Core technology**: Virtualization. *Xen.*
- **Toolkit**: full software stack for a virtualized system
  - *OpenEmbedded* toolchain, Linux kernels, VM filesystems, toolstack, ... . “Batteries included.”
  - Extensible, configurable for diverse use cases and commercial derived products.
  - Supports Service VMs and APIs.
- **Security**: primary criteria in architecture decisions
  - Design enables strong assurances rooted in hardware.
  - Community has expertise.
- **Open Source**: OSI-approved licenses, open development
- **Client** capable: validated support for desktop, laptop hardware and devices
What distinguishes OpenXT?

- **Security** focus of the system architecture and technology selection
- **Community**
  - Depth of expertise in critical technologies, diversity of experience, direction of focus.
  - Members involved in development of platform security technologies across industry.
- **Client** use cases supported
- **Open Source, OpenEmbedded** build and toolchain
  - Participation in upstream communities and commitment to support downstream projects.
- **Validated integration**
  - An integrated system of many complex software components.
  - Derivative products have been successfully Certified and deployed.
Build Product

A current build of OpenXT software today will produce a bootable ISO to run an installer. Install provisions the Xen hypervisor and a collection of helper VMs running Linux to provide a hardened desktop environment for running Windows or Linux desktop workloads.

It provides guest tools with paravirtualized device drivers to install within the guest VMs.

Extensibility of this system:

- At software compile and build time: customizable via OpenEmbedded Layers.
- At deployment time: API support for third-party Service VMs.
Platform Properties of OpenXT

Our community has consensus that these properties of the software are important to us:

- **Integrity of operating software** assured via measured launch and hardware-rooted enforcement of isolation between components.
- **Protection of storage**, confidentiality and integrity of system configuration and user data.
- **Protection of communications**, with isolation between network device control software, encryption software with network credentials, and user software.
- **Protection of concurrent user software** execution environments.
- **Protection of user input** via local keyboard, mouse and touchscreen devices.
- **Protection of graphical output**, with support for high-performance 3D desktops.
- **Policy-based assignment and confinement of hardware peripherals**.
- **Compatibility with modern computer hardware and contemporary operating systems**.
- **Architected and licensed to support commercial derivative software**.

A maintained list of platform properties is at: [https://openxt.atlassian.net/wiki/display/CS/Gov2%3A+OpenXT+Platform+Properties+and+Layers](https://openxt.atlassian.net/wiki/display/CS/Gov2%3A+OpenXT+Platform+Properties+and+Layers)
Architecture aspects / Technologies

- **OpenEmbedded** ecosystem provides toolchain and software platform
- **The OpenXT Project is a strong proponent of disaggregation and isolation**
  - Provides a distributed network stack with VTd-enforced isolation
- **Measured Launch**: using Intel TXT, tboot, TPM 1.2, trousers
- **Read-only root filesystems** with reset upon power operations
- **Linux-based stub-domains** to encapsulate Qemu instances
- **Xen Security Modules** enabled and enforcing
- **SELinux** enabled and enforcing with a tailored policy in Dom0 and NDVM
- **Blktap2** storage path, soon to be blktap3
- **Modern PV-USB** stack with Windows and Linux guest support
- **Specialized human interface device input layer**
OpenEmbedded

A mature Open Source, collaborative, distributed, software packaging project and toolchain.

- OpenEmbedded delivers **many software packages**.
  - Toolkit for assembling an entirely customized Linux distribution.

- We use it to assemble *multiple* customized Linux distributions to build OpenXT
  - Domain 0
  - Stubdomains
  - The OpenXT Installer
  - NDVM
  - UIVM
  - SyncVM
  - ...

OpenEmbedded

OpenEmbedded delivers many software packages.

- Each package is described in a Recipe.
  - Recipes declare package version, source URL, license, build steps, customizations, etc.

- Collections of recipes are maintained in Layers.
  - Software within a layer is maintained together to ensure compatibility between packages.
  - A layer is a git repository with a curated collection of recipe files with community governance.
OpenEmbedded

- Downstream layers can override or extend recipes of upstream layers to apply customizations, if needed.

- Downstream layers benefit from upstream layers component maintainership. **Maximizes community collaboration on shared common components.**

- **OpenXT provides its own Layer.**
  - In near future, it will be multiple layers - we are clustering our components into separate layers.
  - When you work on OpenXT, you work with git repos containing recipes or source code of components that recipes compile and package.

- **OpenEmbedded is one of several projects “upstream” to OpenXT.**
  We work to integrate our changes into the highest layer appropriate.
OpenEmbedded Layers in OpenXT

Layers in OpenXT:

- openembedded-core
- meta-openembedded\(^*\) : meta-xfce, meta-oe, meta-gnome, meta-networking, meta-python
- meta-java
- meta-selinux
- meta-intel
- *coming soon*: meta-virtualization
- xenclient-oe\(^*\)

Other layers of interest:

- meta-security-isafw
- meta-measured
- meta-servicevm

\(^*\) meta-openembedded actually provides multiple layers, rather than just one
\(^*\) "xenclient-oe" should be renamed and separated into "meta-openxt-*" layers
Xen hypervisor and Qemu device emulator

Numerous modifications vs Upstream Xen + Qemu. Motivated by our secure client use cases.

- Suspend, hibernate and power management changes.
- v4v interdomain communication protocol + firewall.
- Modifications to blktap2 storage support for encryption and disk transfers - blktap3 work due to commence soon.
- Cosmetic fixes to correctly display Windows boot graphics.
- ACPI and SMBIOS support for laptop vendor customizations - now upstream and undergoing further development.
- Hardware quirk fixes.
- Setting cache attributes on mapped memory for video support.

The delta vs upstream is decreasing.

Upstream capabilities improve with our involvement and former XenClient features are removed from OpenXT. Current migration work towards to newer hypervisor versions is reducing our patch queue.
Distributed network architecture

A key distinguishing feature of the OpenXT system

Default configuration: All of the physical PCI network devices are assigned to a single Network Device Virtual Machine (NDVM) and isolated using VTd.
Distributed network architecture

A key distinguishing feature of the OpenXT system architecture.

Default configuration: All of the physical PCI network devices are assigned to a single Network Device Virtual Machine (NDVM) and isolated using VTd.

The NDVM contains all the network device drivers. Exports access to the networks via virtual interfaces into the guest VMs. Bridging or routing guest networks and physical networks happens in the NDVM. Dom0 is not part of the guest networking datapath.

User control of networks is via the Network Manager applet in the User Interface VM. This connects to the network-manager daemon via dbus proxied over v4v.

Architecture isolates network device drivers from all VMs and dom0. A compromised network device driver only compromises the NDVM and not the rest of the system.
Network architecture

Alternative configuration

OpenXT also supports running **multiple NDVMs**: one per Network Interface Card.

Configuration instructions are in the project documentation.

This stronger configuration enables **VTd-enforced isolation between physical networks**.

It costs additional resources (RAM, CPU) to run additional NDVMs, and can alter maximum throughput and packet latency across networks but it enables insertion of protected middlebox VMs into the cross-network path.
Network architecture

The OpenXT platform supports development of "NILF-VM"s: **Network Interface Layer Function VMs**. This is to support third-party networking software integration.

eg: a **VPN interposition NILF-VM**. A VM that provides encapsulation of all network traffic via a VPN, interposing in-between a guest Windows VM and the NDVM assigned to the physical device NIC.

This isolates VPN credentials from both the guest Windows software and the network device driver. The unencrypted traffic data is never accessible by the NIC hardware or the network device driver.

**Demonstrates Xen’s strength, the advantage provided by a type-1 hypervisor.**
Core concept:

A sequence of software measurements is taken during system boot, with each component measured prior to launching it, and the measurements securely stored in the hardware Trusted Platform Module.

Measurements are then used to unlock encryption keys that grant access to system configuration and user data, providing confidentiality and constraining access to only when the trusted system software is running.
Measured Launch

- Implementation built upon hardware platform primitives: Intel TXT and a TPM device.
- Measurements include: Xen binary, Dom0 kernel, kernel command lines, Dom0 initrd, Dom0 rootfs, tboot, microcode, Intel ACMs, and XSM policy.
- Uses grub version 2, with a fixed, measured grub command line. Interactive grub prompt is disabled.

The measurements of all components must be correct to acquire the decryption key to unlock access to the host platform configuration data stored on disk and the keys to the disk images of guest VMs.

Notable components:

- Tboot is the Xen launch verifier for TXT.
- TrouSerS, TCG Software Stack. User space software that drives the TPM via a Dom0 kernel device.
- Disk encryption is performed with LUKS. AES-NI accelerates encryption and decryption data paths.
Measured Launch

Recent changes to OpenXT’s Measured Launch:

● LUKS configuration changes to replace a RSA-key wrapped password to an admin supplied password
● Removal of some platform reboots during install for secret sealing
● Preparatory work towards future support for remote attestation
Linux-based Stub-domains

- Motivation: Hardening. Move the large qemu attack surface outside dom0.
  - Has successfully mitigated a number of Xen Security Advisory defects.
  - Acknowledgement: support for disaggregation adds complexity to the hypervisor and tools.
- Limited, read-only root filesystem
- Stateless across domain restarts
- Reduced Linux kernel configuration
Access Control

- **Xen Security Modules**: Enabled and Enforcing
  - Mandatory Access Control within the hypervisor, with the Flask architecture
  - Access control points throughout the hypervisor
  - Governed by a policy loaded at boot

- **SELinux**: Enabled and Enforcing in Dom0 and NDVM
  - Using a tailored policy for confinement
OpenXT PV-USB

- Originally from Virtual Computer’s NXtop, Citrix’s XenClient Enterprise product
  - USB 2.0.
- Linux front-end developed in OpenXT
  - by AIS’s Open Source team. Validated and used in production.
- Orchestrated by a USB control daemon in dom0, also developed in OpenXT
  - Standalone software, primarily interfaces to udev and, to a lesser extent, libUSB.
  - Handles udev events for new devices according to device or device-class policy.
  - Writes XenStore entries to establish connections between the PV-USB front and backends.
OpenXT PV-USB

- Uses a novel two-layer “indirect grant refs” ring structure for efficiency
  - Enables bulk transfer of many grant-refs per ring transaction.

- Upstream-suitable: interest or assistance is very welcome
  - OpenXT is adopting other upstream Xen PV drivers and porting PV-USB to upstream xenbus
OpenXT PV-USB software references

Windows front-end driver code:
https://github.com/OpenXT/xc-vusb
https://github.com/OpenXT/xc-windows

Linux front-end driver code:
https://github.com/OpenXT/pv-linux-drivers/tree/master/xc-vusb

Linux back-end driver code:
Takes control of the assigned USB device’s configurations and interfaces.

USB policy control daemon, runs in the privileged USB control domain, ie. dom0:
Controls USB policy, assignment and management. Monitors udev events, applies policy to the types of devices it sees and assigns devices to VMs by writing XenStore nodes to trigger front-back driver pair to initiate standard xenbus handshake.
https://openxt.atlassian.net/wiki/display/OD/vUSB+Daemon
https://github.com/OpenXT/vusb-daemon
Secure Keyboard and Mouse Input

An essential component for client virtualization. The input server runs in Dom0.

- Monitors udev for new input devices
  - Claims HID devices: keyboard, mouse, touchpad, touchscreen
- Demultiplexes input events
- Directs input events to the intended recipient VM
  - Including the UIVM, showing the local console UI
  - Prevents input snooping by other VMs
  - Enforces platform screen lock
  - Performs secure password capture for Dom0 : UIVM cannot snoop.
- Scales mouse movement for guest VMs running at different resolutions
v4v: an interdomain communication transport

- An OpenXT technology, originally developed for XenClient.
- Hypervisor-mediated data copies via private ring buffers with notifications.
- Used by OpenXT and in production in its derivative products, plus a variant in use at Bromium.
- Has benefitted from previous reviews by the Xen Community.
**v4v**: an interdomain communication transport

Motivations for v4v versus any other interdomain communication mechanism:

- **Strong isolation** between communicating domains  
  - No memory is shared between VMs
- **Strong enforcement of policy controls** on VM communication  
  - A firewall within the hypervisor enforces rules that are set externally
- **High performance** suitable for sustained throughput
- **A clean mapping to Linux and Windows native I/O primitives**
- **Clear separation from guest Operating System networking stacks**
- **A foundation for the future work** that we intend to do
**v4v**: an interdomain communication transport

**Known limitations:**

- **Current implementation has exposure to resource exhaustion**
  - To be remedied with consumption constraints
  - DoS and scalability support were lesser concerns for the original client use cases.

- **v4vtables firewall is a basic implementation**
  - A new firewall is proposed to replace or extend v4vtables with XSM/Flask and support for in-guest SELinux

- **Linux driver software requires improvement**

- **libvchan bindings have been requested**

- **Documentation is required**
  - v4v comparison with other interdomain communication technologies
  - Hypervisor hypercall interface, Linux and Windows v4v driver interface and user-space library interface
  - Mechanisms to constrain v4v: XSM and the v4vtables-successor
v4v: an interdomain communication transport

Strong isolation avoids these concerns that affect inter-VM shared-memory technologies:

Impact on protocol integrity
- Peer domains can directly tamper with control structures in the shared memory region.

Impact on peer authentication
- Cannot directly know which principal wrote the protocol payload into memory to authenticate sender.

Impact on confidentiality
- Data may intentionally or accidentally leak through parts of shared memory not in use by the protocol.
- Page ownership may be intentionally or accidentally shared with additional domains, allowing observation of data intended for one receiver.

Cost of extra assurance dependencies
- Protocol connection via XenStore forces XenStore into the TCB for communicating components.
**v4v**: an interdomain communication transport

**Upcoming work:**

- **New firewall using XSM/Flask architecture**
  - Work towards a unified access control architecture in OpenXT with clear policy representation, better tooling, fewer opportunities for gaps or error
  - XSM/Flask to enforce Mandatory Access Control over v4v connections between domains
  - v4vtables may be extended to enable guests to express self-protection rules
  - SELinux to control v4v bind, send operations by individual processes
  - Add support for obtaining Flask peer labels for v4v, as per event channels
  - Convey SELinux security context across domain boundaries for access checks at recipient

- Address the known limitations described in the earlier slide.

We continue to be interested in engaging upstream Xen with v4v.
Recent Developments

- **OpenXT Summit**: June 2016, Washington, D.C.
  - 48 attendees from 24 organizations
  - Two days of presentations and moderated discussions on ecosystem topics

- **Version 6.0 of OpenXT software release**
  - Update the core distribution to OpenEmbedded Jethro release
  - Skylake hardware platform support
  - Xen hypervisor upgrade to 4.3.4 - *Xen 4.6.1 is now in our master branch; work on 4.7 is next*
  - Linux kernel upgrade to 4.4.16 - *OpenXT 6.0.x point releases will track 4.4.x kernel updates*
  - Qemu build configuration security hardened
  - Codebase security review. Package updates and selective patch inclusion for security fixes
  - Containerized build system, improved ease of use

- **Project Governance**
  - Project charter documents and codifying project processes
Work in Progress

Major focus is on accelerated tracking and engagement with upstream projects.

- **Restructuring OpenXT into distinct OpenEmbedded Layers**
  - Simplifies work for derivative projects to choose just the components they need
  - Optionality addresses the tension between feature addition vs. increase in attack surface

- **Reducing specialized components** of the OpenXT software
  - Adopting versions present in upstream OpenEmbedded layers
  - Working with upstream layers to add capabilities we need, assist with maintenance
  - Shrinking OpenXT Project codebase to just that which is unique to OpenXT requirements

- **Xen hypervisor upgrade**
  - 4.6.1 has just been successfully tested and merged into master development branch
  - 4.7.x will be immediate next target
Work in Progress

- **Toolstack refactoring**
  - Port to introduce libxl into the OpenXT Haskell toolstack running outside dom0

- **v4v improvement**
  - Please come and talk to community members about this

- **Measured Launch enhancements**
  - Towards forward-sealing across software upgrades

- **Adopting upstream Windows PV-drivers**
  - For network and block devices, to use in addition to our PV drivers for other devices

- **Alternative graphics display architecture**
  - Developed on an OpenXT-derivative platform
  - Please see the AIS presentation at this Xen Developer Summit
Roadmap items

- Community members have plans to integrate Xen’s vTPM and vTPM manager into OpenXT in the near future.
- OpenXT community intends to adopt and maintain blktap3, working with the XenServer team to assist support of their existing customer requirements.
- Make tboot into a UEFI module.
Where we work on OpenXT

- Public mailing list: openxt@googlegroups.com
- IRC #OpenXT on freenode
- Monthly community telephone call, open to all
- Confluence public wiki
- JIRA issue tracker on Atlassian cloud
- Presence in upstream projects: OpenEmbedded, Xen, Linux TPM tools, ...

You are welcome to join us!

openxt.org
Thank-you for your attention.

Thanks to the members of the OpenXT community for the work described here and material supporting this presentation.